МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Ярославский государственный университет им. П.Г. Демидова»

Математический факультет

Кафедра компьютерной безопасности

Заведующий кафедрой:

д. ф. - м. н., профессор

Мурин Д. М.

Направление подготовки

10.05.01 Компьютерная безопасность

Курсовая работа

Обнаружение аномальной активности в сети на основе анализа статистических отклонений в работе системы.

Научный руководитель

д. ф. - м. н., доцент

\_\_\_\_\_\_\_\_ Д. А. Савинов

«\_\_» \_\_\_\_\_\_\_\_\_ 2025 г.

Студент группы КБ-51СО

\_\_\_\_\_\_\_\_ А.Ю.Сластухин

«\_\_» \_\_\_\_\_\_\_\_\_ 2025 г.

Ярославль 2025 г.

Оглавление

Оглавление 2

Введение 3

IDS/IPS системы 5

Классификация (По области применения) 5

Классификация (По принципу действия) 5

Сигнатурные 5

Аномальные 6

Сферы применения поиска аномалий. 7

Подходы к выявлению аномалий. 8

Реализация собственной IDS системы 9

Сетевой портрет 9

Проблема изучение аномалий в поведении человека-системы 10

Постановка задачи 10

Архитектура анализатора 11

Базовый анализатор 12

Метрика оценка качества модели 14

Алгоритм корректировки весов 14

Алгоритм подбора весов 15

Результат 15

Применение при решении задачи выявления аномалий для датасета TelecomX 16

Описание 16

Уточнения 17

Выбор модели данных, предобработка грязных данных. 17

Конфигурирование анализаторов 18

Обучение и анализ 18

Анализ аномальной активности пользователя 5828 19

Анализ с точки зрения злоумышленника 21

Итог по TelecomX 22

Итоги 22

Что дальше? 22

Литература 23

Введение

В последние десятилетия, в связи с активным развитием технологий, в частности, персональных компьютеров и сети интернет, у человека появился доступ к огромному океану различной информации, многие проблемы стали решаться в несколько кликов мыши. Это привело к тому, что и злоумышленники получили в своё распоряжение отличные инструменты для совершения зловредных действий.

Например, с 1960-х до середины 1980-х телефонные хулиганы часто звонили из телефонных автоматов, в пределах ограниченной местности, с учётом того, что себестоимость услуги компании была близка к 0, а число подобных инцидентом было мало, телефонные компании не теряли в прибыли насколько, что могли игнорировать проблему. Или подделка долларовых банкнот: спецслужбы США выяснили, что в руки злоумышленников попало определённое число старых печатных машин, способных печатать доллары США. Оценили общее число машин и их производительность и пришли к выводу, что общий ущерб является каплей в море. А теперь представьте, как с развитием технологий всё изменилось – достаточно одному злоумышленнику придумать способ, даже неэффективный, для получения 1 цента и выложить программу в интернет – финансовая система может быть уничтожена, поскольку технические возможности распределённой системы злоумышленника оцениваются сверху всеми устройствами в сети.

Природа атак не изменилась с развитием техники, старые подходы, способы обмана и хищения используются при новых инструментах, но старые методы защиты частично перестали работать из-за возросших требований. Мы так же пытаемся обезопасить свои личные устройства или корпоративные сети, но проблема в том, что на любое средство защиты найдётся способ его обойти.  
  
Один из подходов к решению возникших проблем, помимо защиты сети в целом с помощью различных брандмауэров и фильтров, будем рассматривать защиту отдельно взятых устройств - это подобно личной гигиене, отсутствие которой в средние века выкашивало целые поселения – так и сейчас, защитив отдельно взятые устройства, мы можем значительно снизить ресурсы сети, которые могут быть взяты под контроль, сможем снизить потенциальный ущерб.

Среди способов защиты традиционно выделяется статический анализ – кто-то когда-то уже подвергся нападению и выявил уязвимость, оповестил остальных и системы скорректировали. Эффективность этого подхода в том, что большинство инцидентов проходит мимо нас, но мы перекладываем ответственность на кого-то другого, кто сможет выявить угрозу и всех оповестить, считаем, что наши устройства никому не нужны – идейно, это работает неплохо, но существует несколько проблем:

* новая угроза может быть не выявлена статическим анализом
* кто-то не оповестит других, посчитав, что найдётся более опытный специалист, кто справиться с проблемой за него – перекладывание ответственности

и угроза за короткое время распространиться по сети, нанесет ущерб, как обычным пользователям, так и крупным компаниям.

Поэтому выделяем альтернативный подход – основанный на следующей идее: Различные пользователи обладают своим характером, потребностями, целями, знанием и опытом, решают свои задачи, например, увлекаются выпечкой или разрабатывают приложение на Python - так и компьютерные системы крупных компаний тоже обладают своими особенностями – занимаются массовыми рассылками, при этом не проводят сложных вычислений, какие-то наоборот только считают, и общаются только с выделенными сетевыми узлами. Согласитесь, будет странно, если почтовый сервис начиная с какого-то момента будет выделять много процессорного времени на решение одной задачи, когда он обычно решает много маленьких. Вы мне скажете, что с людьми это не работает – разработчик может заинтересоваться выпечкой – конечно, может, но изменения в жизни и характере человека чаще происходят постепенно, и если происходят стремительно, то их нельзя оставлять без внимания. В случае человека подобная система, скорее всего, не заметит изменений, т.к. они будут постепенны. В случае приложений или устройств – анализатор аномалий можно будет обойти, если вносить корректировки в активность плавно, но на это уйдёт немало времени, которое позволит выиграть время, более того потребуется чётко выстроить постепенные изменения – в каждом конкретном случае это отдельная, непростая задача.

Получаем, что нужно использовать статический анализ совместно с анализом аномалий – статический защитит от известных угроз, а основанный на аномалиях выявит потенциально опасное действие или выиграет время на обнаружение зловреда.

О том, какие подходы к анализу аномалий существуют, какие проблемы возникают, эффективны ли эти подходы поговорим в рамках курсовой работы.

IDS/IPS системы

IDS (Intrusion Detection System) - система обнаружения вторжений.

IPS (Intrusion Prevention System) - система предотвращения вторжений.

IDS является сложным, комплексный решением, которое может сочетать в себе, анализ файлов, как это делает антивирус, контроль трафика и работу с правилами сетевого взаимодействия, как это делает файрвол.

Классификация (По области применения)

* Network Intrusion Detection System (NIDS) - контролирует работу в сети, взаимодействие нескольких хостовых узлов.
* Host-based Intrusion Detection System (HIDS) – работает на конкретном хосте, не имеет доступа к другим узлам.

NIDS по своей идее очень похожа на файрвол, просматривает весь входящий и исходящий трафик, но является более мощным инструментом, поскольку может находить глубокие признаки аномальной активности.

Основным минусом является тяжеловесность – для глубокого анализа на наличие угроз, найденных в взаимодействии различных узлов, требуются значительные вычислительные мощности, сопоставимые с мощностями всей системы, а часто и превышающими её, поскольку часто требуется обработать всё, что происходит в сети.

Часто NIDS нацелены именно на анализ сетевого взаимодействия в то время, как HIDS анализируют всё, что происходит на конкретном узле, включая трафик, файловую системы, журналы событий – таким образом, NIDS ставят на всю сеть, а на критически важные точки ставят HIDS.

Дополнительно выделяют:

* PIDS (Perimeter Intrusion Detection Systems) - подобно вайерволу контролируют только периметр сети, а не всю сеть.
* VMIDS (Virtual Machine-based Intrusion Detection Systems) – принципиальным отличием является то, что устанавливается на устройство, а на виртуальную машину, что позволяет быстрее её переносить.

Классификация (По принципу действия)

Сигнатурные

Анализируют текущее состояние системы на предмет нахождения уже известных сигнатур угроз. В этом отношении такие системы пересекаются с антивирусами, имеют минус – не реагируют на новые уязвимости. Поэтому требуется своевременное получать информацию о новых уязвимостях.

*Сигнатурой* называется шаблон уже известной угрозы.

Помимо сигнатур выделяют *состояния*: в начальный момент времени считаем, что система в безопасном состоянии, но каждое действие в системе: несанкционированное действие пользователя, сетевое обращение, установка и активация ПО, особенно драйверов, может привести в скомпрометированное состояние. Очевидно, что NIDS часто отслеживают сигнатуры, а HIDS состояния.

Аномальные

*Аномалией* является любое действие, которое хоть по каким-то признакам отличается от нормы, что даёт возможность предполагать, что система могла перейти в скомпрометированное состояние.

Аномалии делятся на:

* Статистические – сопоставляют профиль штатной работы с текущим, используют статистические алгоритмы. Например, в текущем профиле повышена нагрузка на сеть или диск, число операций, интервал между ними – снижены или наоборот завышены. Всё, что указывает на нарушение штатной работы.
* Поведенческие – части системы пытаются выполнять функции, которые им не предусмотрены или нарушается порядок выполнения, например, есть приложение пытается отправить результат до того, как задача была выполнена, или же штатный протокол используется не по назначению.
* Аномалии трафика – статистически трафик может быть штатным, но в нём могут присутствовать нетипичные особенности.

Аномалии могут иметь черты разных типов – это может быть аномалия трафика, в которой помимо угрожающей активности может присутствовать нарушение профиля штатной активности.

В рамках этой работы особый интерес для нас представляют именно системы, работающие с аномалиями. Они более сложные, разнообразные, наукоёмкие с точки зрения используемых подходов, тяжеловесные с точки зрения требуемых ресурсов.

Системы поиска аномалий способны помочь выявить новую угрозу, в то же время они не дают никакой гарантии, что в принципе сработают, в то время как сигнатурные гарантируют, что конкретные сценарии практически недостижимы.

Часто под аномальными представляют нейросети, их действительно можно применять, но только ими дело не ограничивается – подойдёт любой алгоритм, возвращающий ответ Да/Нет/Не знаю.

Сферы применения поиска аномалий.

Прежде, чем изучать способы выявления аномалий, рассмотрим несколько сфер возможного применения выявления аномалий:

1.*Анализ штатной активности уже работающих систем* - в 2010 году сетевой червь *Stuxnet* атаковал иранское предприятие, изменив настройки оборудования, привёл к преждевременному выходу из строя центрифуг для обогащения уранового топлива. Есть веские причины полагать, что вирус был разработан при сотрудничестве США и Израиля с целью саботировать иранскую ядерную программу – если бы система контроля учитывала штатную активность оборудования, экономических и стратегических потерь можно было бы избежать.

2.*Выявление аномальных потребительских паттернов или мошеннических действий* – пользователь Боб периодически совершает переводы Алисе. В случае если Боб попытается перевести Алисе сильно больше, чем он делает обычно или попытается сделать перевод в другую страну, хотя обычно он совершает переводы в рамках своего города – банковская система должна отреагировать и запросить дополнительное подтверждение или заблокировать операцию.

3.*Рекомендательные системы* – бывает очень важно отвечать на вопрос, понравится ли пользователю той или иной продукт, опираясь на профиль его активности – какие категории товаров он смотрел в последнее время, что купил –важно предоставить пользователю связанные товары: купил человек подгузники, вероятно у него есть ребёнок, можно в ленту добавить детское питание, одежду, игрушки.

Возникает вопрос – при чём тут аномалии? А при том, что рекомендательная система может быть преобразована в детектор аномалий.

Например:

* штатная активность пользователя – это основа для рекомендательной системы, аналог просмотрам и покупкам.
* новое действие пользователя – это то, чему система даёт оценку.
* результат работы – если действие может считаться рекомендованным, то штатный профиль активности не нарушен, иначе это аномальная ситуация.

Возникает вопрос, что первично, а что вторично – сложный вопрос, скорее всего, это одна и та же сущность, рассмотренная под разные сферы применения.

Отмечу, что практические вопросы, возникающие при построении рекомендательных систем, такие же трудные, как и проблемы построения систем выявления аномалий – почва для глубоких исследований эффективных подходов.

Подробнее рассмотрим позже.

Приходим к выводу, что поиск аномалий – важная задача.

Подходы к выявлению аномалий.

Важно понимать, что выявление аномалий нетривиальная задача – существует множество алгоритмов, но в большинстве они используют методы, которые описаны ниже.

В этом году, наверное, ограничусь тем, что перечислю

1.Метод *опорных векторов* с одним классом One-Class SVM

(!) Расписать подробнее

Подходит, когда в обучающем наборе данные подчиняются нормальному распределению, а в тестовом содержат аномалии.

Считается, что этот подход наиболее часто применимый при поиске аномалий.

2. Метод *изолирующего леса* – isolate forest

(!) Расписать подробнее

Опирается то, что при «случайном» построении деревьев выбросы будут попадать в листья на ранних этапах (на небольшой глубине дерева), т.е. выбросы проще «изолировать». Выделение аномальных значений происходит на первых итерациях работы алгоритма.

3. *Elliptic envelope*  и другие статистические методы.

4. *Метрические* методы

5. *Кластерные* методы

6. Метод *главных компонент*

7. Методы на базе *прогнозирования временных рядов* – если рассматриваемое значение выбивается из временного ряда, то значение считается аномальным.

8.Обучение с учителем – регрессия и классификация

9.Методы на базе алгоритмов решения задач рекомендаций.

Решение задачи поиска аномалий часто требует индивидуального подхода к конкретной проблеме – это искусство, требующее своевременного применения различных знаний.

Более того, доказано, что не существует универсального решения для борьбы с вирусами, сетевыми угрозами – для любой существующей системы можно предложить подход, позволяющий обойти алгоритмы детектирования. Таким образом различные методы применяются совместно – группируются, в зависимости от системы.

Пример: анализаторы сетевого трафика Positive Techologies представляют целые цепочки анализаторов, построенные для максимально эффективного использования вычислительных ресурсов – не забываем, что решение задачи поиска угроз требует затраты, сопоставимые с содержанием такой же системы.

Реализация собственной IDS системы

Прежде, чем приступить к реализации обсудим важные понятия и возможную архитектуру подобных систем.

Я буду придерживаться подхода рассмотрения системы как человека, опираясь на то, как мы работает с людьми, чтобы лучше понять, как устроено наше человеческое мышление.

Сетевой портрет

Представьте, что система подобно человеку имеет некоторые поведенческие особенности, по которым её можно отличить от другой системы.

При этом эти особенности не обязательно уникальны – множество систем может им соответствовать.

Замечание: как отпечаток пальца человека не является его поведенческой особенностью, так и имя или адрес системы не являются её поведенческой особенностью, это уникальный идентификатор, позволяющий выделить среди остальных, поэтому мы будем анализировать именно действия системы.

Вспомните хорошо известного вам знакомого, попробуйте сгенерировать действие, узнав о котором, вы бы не удивились, что его сделал именно этот человек. Представьте, действие, узнав о котором, вы бы засомневались в его правдивости.

Как же вы рассуждали?

Предположу, что вы взяли все известные вам действия этого человека, из которых сложили его портрет и сопоставили полученный портрет с действием, отвечая на вопросы соответствия портрета действию ответами – «Да», «Нет», «Не знаю».

Чем лучше, вы знаете человека, тем точнее будет ваш ответ.

С системами аналогично, предполагаю, что анализируя активность системы, можно построить её портрет, который позволит нам выявлять аномальную активность.

Возникаю следующие проблемы:

1. Не хватка информации или её противоречивость – портрет смешанный, это может негативно сказаться на чёткости портрета.
2. Возможность множеством последовательных незначительных изменений исказить исходный портрет, при этом каждое изменение может укладываться в доверительный интервал, и мы не узнаем об аномальном преображении.
3. Проблема реагирования на незначительные изменения – корректировать портрет его или проводить расследование, по итогам которого принять действие как аномальное и не заносить его в портрет – вопрос применяемой политики, может сложиться так, что изменения соответствуют изменениям владельца системы.
4. Ложные срабатывания – если система будет реагировать на незначительные изменения, которые допустимы, система анализа потеряет доверие, к её выбросам не будут прислушиваться.

Как мы видим большинство проблем возникают из-за недостоверных данных или неправильной политики, подобрать которую оптимально представляется возможным в индивидуальном порядке.

Проблема изучение аномалий в поведении человека-системы

Заметим, что промышленными системами ситуация проще, ведь они более инертны и предсказуемы – если почтовый сервер стал выполнять тяжёлую вычислительную задачу вместо множества маленьких – это повод расследовать.

С людьми тяжелее, они менее предсказуемы, а системы, которые они используют отражают их индивидуальные особенности – поэтому важно, понимать, как изучать людей, чтобы изучать деятельность их систем, в свою очередь выявление аномалий в действиях системы может помочь выявить негативные аномальные изменения в человеке, это может быть интересно в целях обеспечения безопасности.

Именно на проблеме выявления аномалий в деятельности человека-систем я и буду делать акцент в текущей работе. Поскольку продвинемся в решении сразу нескольких задач – обеспечения безопасности как отдельных узлов сети, так и всей сети целиком, так и изучение человеческого мышления.

Постановка задачи

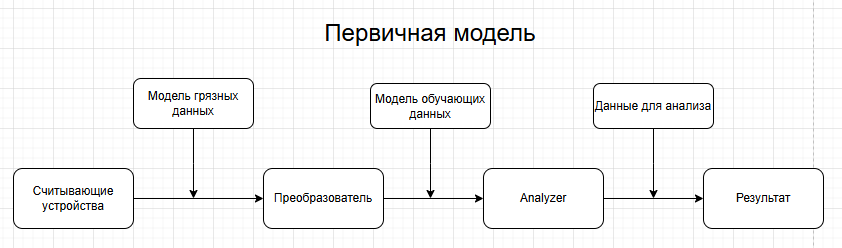
Цель: построить универсальную систему поиска аномали.

Под *универсальностью* понимаем следующее: пользователь, уровня продвинутого разработчика или системного администратора получит инструмент, для работы которого потребуется передать модель данных DataFrame в формате .csv или .txt.

По входной модели потребуется настроить анализаторы (Analyzers).

Analyzers, как сущности предопределены, их потребуется выбрать и настроить параметры, от которых будет зависеть, насколько резко система будет реагировать на изменение сетевого портрета.

Архитектура анализатора



1.Считывающие устройства – оборудование, коммутаторы, маршрутизаторы, перехватчики трафика, например, WireShark и т.п.

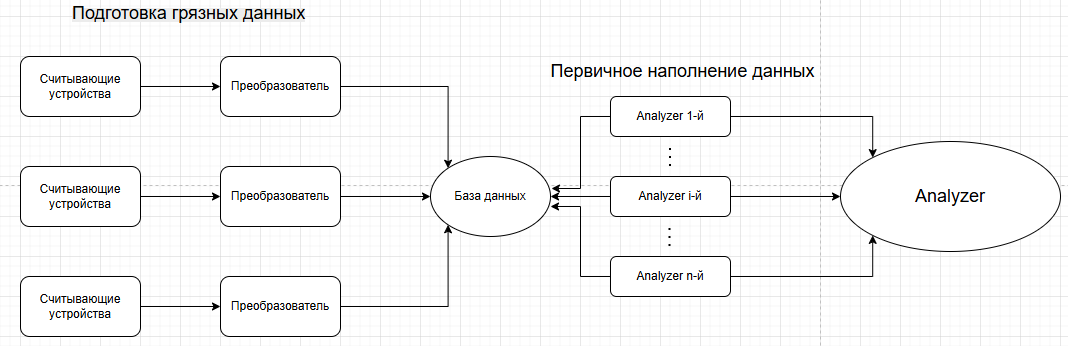
2.Модель грязных данных – полученные при считывании с устройств, часто это разрозненные данные (смешанные, разных форматов), непригодные для обучения.

3.Преобразователь – преобразует грязные данные в данные, пригодные для обучения.

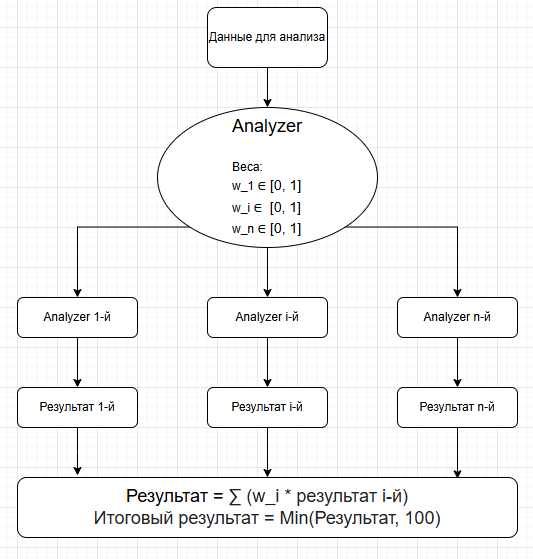
4.Модель обучающих данных – подготовленные, сгруппированные, одного формата.

5.Analyzer - устройство по модели чёрного ящика, которое каким-либо образом учитывает обучающие данные, и способно выдавать результат для новых данных.

Замечание: Analyzer может быть представлен комбинацией анализаторов.



Анализ отражается следующей схемой:



Каждый из анализаторов независимо от других предоставляет результат ∈ [0, 100]

Результат – это уровень доверия, если он 100, то анализатор не считает данные аномальными.

У каждого анализатора есть вес в принятии общего решения .

Замечание: в данном анализаторе нет ограничения .

Возможны варианты, когда оно есть, но в данный момент считаю, что разные комбинации анализаторов на разных данных могут дать положительный результат, поэтому это ограничение отбрасываю – это место для исследования, считаю, что в будущем ограничение стоит принять и пересмотреть архитектуру.

Базовый анализатор

В построенной библиотеке есть предопределённые базовые анализаторы, которые можно использовать, как для анализа данные, так и для расширения – для создания собственных анализаторов за счёт наследования от базовых.

Базовый анализатор имеет следующий минимальный интерфейс:

1. public bool **Filter**(DataFrameRow row)– метод фильтрации, позволяет отсеять данные, которые непригодны для обучения этим анализатором.

Например, строковый анализатор ничего не может сказать по пустому значению.

1. public PredictInfo **Estimate**(DataFrameRow row) – метод предсказания.

**PredictInfo** – обёртка для результата, включает в себе помимо результата, используемые анализаторы и обоснование оценки, для удобства анализа.

1. public void **LoadRow**(DataFrameRow row) – метод внесения достоверных данных.

Каждый анализатор может быть представлен логической комбинацией других анализаторов:

* группой, где у каждого есть коэффициент влияния,
* комбинацией – для того, чтобы анализаторы можно было использовать только совместно или использовать по правилу ИЛИ с одним общим весом.

Но в общем, каждый анализатор может рассматриваться как отдельный, независимый от других, со скрытой реализацией, главное, что они соответствовали общему интерфейсу.

Замечание: Успех использования библиотеки зависит от того, как в каждой конкретной задаче будет выстроен используемый анализатор – универсального решения нет, это большая задача, наравне с тем, какую модель входных данных использовать – алгоритм подберёт оптимальные веса, но от разработчика требуются осмысленные решения.

На текущую версию доступны следующие базовые анализаторы:

1. **IdAnalyzer** – идентификаторы, самый простой анализатор.
2. **ArrayAnalyzer** – позволяет группировать идентификаторы, например – идентификаторы продуктов в чеке или список получателей сообщения.
3. **NumberAnalyzer** – числовые значения, тк числа непрерывны, то хранить их может быть сложно, поэтому подвергаются квантованию по уровню, что позволяет эффективно хранить и использовать на новых, не встречаемых ранее, значениях.
4. **StringAnalyzer** – строковые значения, из строки выделяются корни слов, сортируются и рассматриваются как ArrayAnalyzer – подход позволяет эффективно хранить, но из-за сортировки и отброса частиц, таких, как НЕ, теряется смысл исходного значения.

Текущие базовые анализаторы примитивны – все построены на эффективных словарях, работа с которыми инкапсулирована в сущности **History** (попытка воспроизвести примитивную модель человеческой памяти), но это обеспечивает высокую скорость работы и низкие размеры моделей.

В будущем планируется расширить базовый набор за счёт использования алгоритмов машинного обучения, многие из которых указаны выше.

Метрика оценка качества модели

Опр: **RSquared** (Коэффициент дитерминации) – метрика для оценки качества модели, принимает значение от нуля до единицы.

Модель можно считать *неплохой*, если коэффициент больше 0.5, можно интерпретировать как соответствие модели данным. Чем ближе значение коэффициента к 1, тем сильнее соответствие.

* Для *приемлемых* моделей предполагается, что коэффициент детерминации должен быть хотя бы не меньше 50 % (коэффициент корреляции превышает 70 %).
* Модели с коэффициентом детерминации выше 80 % можно признать *хорошими* (коэффициент корреляции превышает 90 %).
* Значение коэффициента детерминации 1 означает функциональную зависимость между переменными.

Алгоритм корректировки весов

Вход:

* expected, current - ожидаемый и фактический результат,
* weights – текущий набор весов.

Выход:

* new\_ weights – скорректированный набор весов.

В нашем случае алгоритм подбора весов реализуется, как

1. Вычислить общую ошибку, как разность ожидаемого результата и текущего:
2. Вычислить значение сигмоиды:
3. Вычислить значение корректировочного коэффициента:

Параметр отвечает за скорость, с которой вес будет скорректирован. Параметр перебирается в процессе тренировки, тк в зависимости от обучающей выборки он может быть разным.

delta – коэффициент для корректировки общего анализатора, но он состоит из нескольких анализаторов – требуется распределить корректировочное значение.

1. used\_analyzers – анализаторы, которые принимали участие в полученном результате.

Анализатор может дать ответ («не знаю»), в таком случае он не учитывается в корректировке весов.

1. Корректировочное значение для веса анализатора:
2. Новое значение веса анализатора:

Алгоритм подбора весов

Этап 1 (Инициализация):

1. Разбивается общий набор обучающих данных на тренировочные (30%) и эталонные (70%).
2. Эталонные данные загружаются в анализатор.

Этап 2 (Подбор весов):

1. Генерируется случайный вес для каждого анализатора – мы изначально не знаем, какой анализатор лучше отражает особенности исследуемого датасета, но в процессе корректировки весов выявляется эта зависимость.

Если анализатор бесполезен, то его ни разу не используют для корректировки весов или его вес окажется мал по результатам тренировки.

Замечание: Думаю, в последствии откажусь от случайной генерации начальных весов – будто это не вносит никакой пользы, это рудимент первой версии.

1. Считается значение качества модели RSquared.
2. Цикл, по тренировочным данным:
   1. Предсказывается результат для текущей записи ([0, 100+]).
   2. Срабатывает алгоритм корректировки весов.

Этап 3 (Повторение этапа 2):

В частности, перебор шага обучения learn\_step и попытка сгенерировать более подходящие начальные веса.

Этап 4 (Сохранение наилучших весов)

Результат

Построена библиотека, позволяющая сконфигурировать решения задачи выявления аномалий на произвольном датасете.

Ключевый фактором успеха является решение задач:

1. Сформить модель данных для анализа – если она не будет коррелировать с реальными аномалиями, то система их не сможет найти.
2. Сконфигурировать систему анализаторов из предложенных базовых.

Решение описанных задач требует индивидуального подхода, поэтому возлагаются на разработчика, который будет внедрять систему под свои задачи. Система в будущем должна быть удобной и значительно автоматизировать многие рутинные процессы, связанные с решением поставленных задач – например, предложить автоматическую конфигурацию по произвольному датасету, которую в последствии нужно будет только скорректировать.

Рассмотрим применение библиотеки на примере датасета TelecomX

Применение при решении задачи выявления аномалий для датасета TelecomX

Описание

В качестве датасета буду использовать датасет, сгенерированный для хакатона по обработке больших данных в 2024 году.

Постановка задачи:

«С линейного оборудования узлов связи раз в какое-то время (например, раз в 10 минут) снимаются логи, содержащие информацию об интернет-соединении абонентов:

* номер сеанса связи,
* дата начала,
* дата конца,
* продолжительность сеанса,
* номер абонента,
* количество скачанного и переданного им трафика.

Когда данные прилетали на хранилище, они обрабатывались аналитическим pipeline для выявления аномалий потребления трафика абонентами за последнее время.

В частности, если характер потребления трафика кардинально изменился в сторону увеличения потребления по сравнению с предыдущим *за аналогичный период*, то есть подозрение на то, что оборудование абонента взломано и превратилось, например, в узел DDOS-сети или в спам-сервер.

Такой pipeline запускался каждый час, и результатом его работы была плоская таблица — витрина данных, в которой представлен список работавших в последний час абонентов (с их контактными данными) и их предполагаемый статус: was hacked или нет.»

Кейс построен на основе реального опыта специалистов.

В данный момент создано три набора:

* Telecom10k — телеком-компания с 10 000 абонентов, 1 млн записей, 51 Мб данных.
* Telecom100k — телеком-компания с 100 000 абонентов, 11 млн записей, 688 Мб данных.
* Telecom1000k — телеком-компания с 1 000 000 абонентов, 117 млн записей, 7,2 Гб данных

Уточнения

* В своей реализации не учитываю аналогичные периоды, тк это задача анализа временных рядов, она пока в разработке.
* Вместо анализа последнего часа проанализирую несколько часов с целью выявить устойчивое отклонение в работе системы.
* Весь процесс анализа, включающий предобработку инкапсулированы в сущности **TelecomX**.

Выбор модели данных, предобработка грязных данных.

На вход нам приходит «грязный» обьём данных – каждое оборудование, снимающее логи, передаёт данные в необработанном виде, имеется информации о сессиях различных пользователей в течение 10 минут.

Для составления портретов активности нужно провести агрегацию данных по каждому пользователю в разрезе каждого часа.

Модель для анализа имеет следующий вид:

1. Пользователь
2. Дата анализируемого часа
3. Число сессий
4. Переданный трафик
5. Полученный трафик
6. Продолжительность

Пункты 4-6 особенные, тк при агрегации мы должны выбрать подход:

* Общая сумма по всем сессиям
* Медианное значение
* Максимальное значение

Пункты 1-2 будут нести информационный характер, в процессе анализа не учитываются.

С учётом того, что алгоритм позволяет подобрать оптимальные веса, то будем использовать параллельно все подходы для пунктов 4-6, предполагая, что для каждого пользователя может играть роль своя метрика.

По итогу предобработки для каждого пользователя сформирован свой файл .csv, в котором хранятся сущности, готовые к анализу.

Конфигурирование анализаторов

С учётом того, что анализируемые метрики (Число сессий, продолжительность, объём переданного и полученного трафика) – числовые, то все анализаторы будут иметь тип **NumberAnalyzer**, настройки по умолчанию, составные анализаторы не применяются.

Обучение и анализ

Для каждого пользователя по отдельности на его данных обучается портрет активности. Процесс обучения включается погрузку эталонных данных и автоматический побор весов для каждого анализатора.

Результатом является портрет (общий анализатор), определяет уровень доверия к действиям пользователя.

Замечание:

При текущей реализации на базе эффективных кэшей есть как плюсы, так и минусы.

При необходимости дозагрузить новые данные в портрет не требуется обязательного перераспределения весов – хотя, конечно, периодически это делать стоит.

При такой реализации требуется решить задачу откидывания «странных», редких данных в обучающем датасете – это позволяет выделить скелет, основные черты пользователя, но портрет потеряет детали (меткие черты тоже важны).

Отбрасывание частично решает проблему, но это потенциальная точка для эксплуатации злоумышленником, при этом, использовать все имеющиеся данные так же опасно – позже увидим, почему.

В текущей реализации «странные» значения не отбрасываются автоматически.

Обучив модели 635 пользователей, и проанализировав статистику их трафика за последние 6 часов, удалось выявить 3 пользователя с устойчивыми аномалиями – за 6 часов, модель зафиксировала более 2х аномалий, т.е. более 50% времени за последние 6 часов, активность была аномальной.

Анализ аномальной активности пользователя 5828

Сначала приведены графики изменения статистических показателей.

Жирными точками выделены значения, где была зафиксирована аномальная активность – это не означает, что аномалия на конкретном графике, а лишь указывает на то, что в это время система зафиксировала аномалию, графики отражают изменение показателей.

Как эти показатели используются системой, графики не отражают.

В конце приведён график, на котором показан уровень доверия к действиям конкретного пользователя

Пользователь 5828 (зафиксирована аномалия):

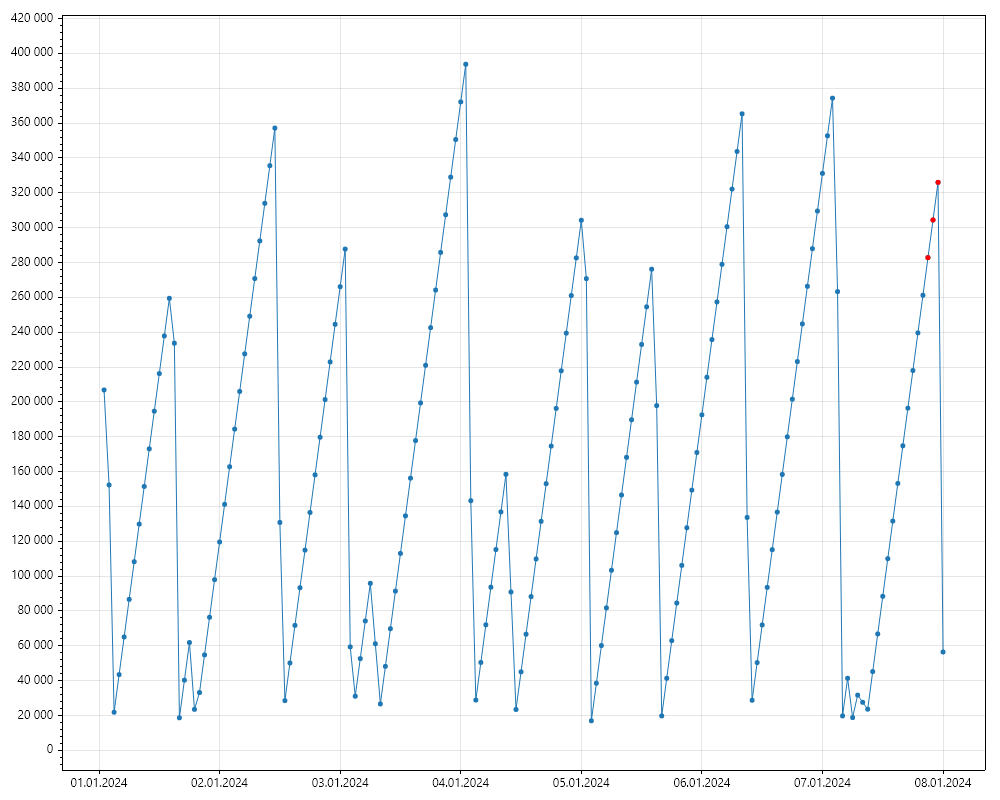


Рисунок 1 общая продолжительность сессий пользователя 5828

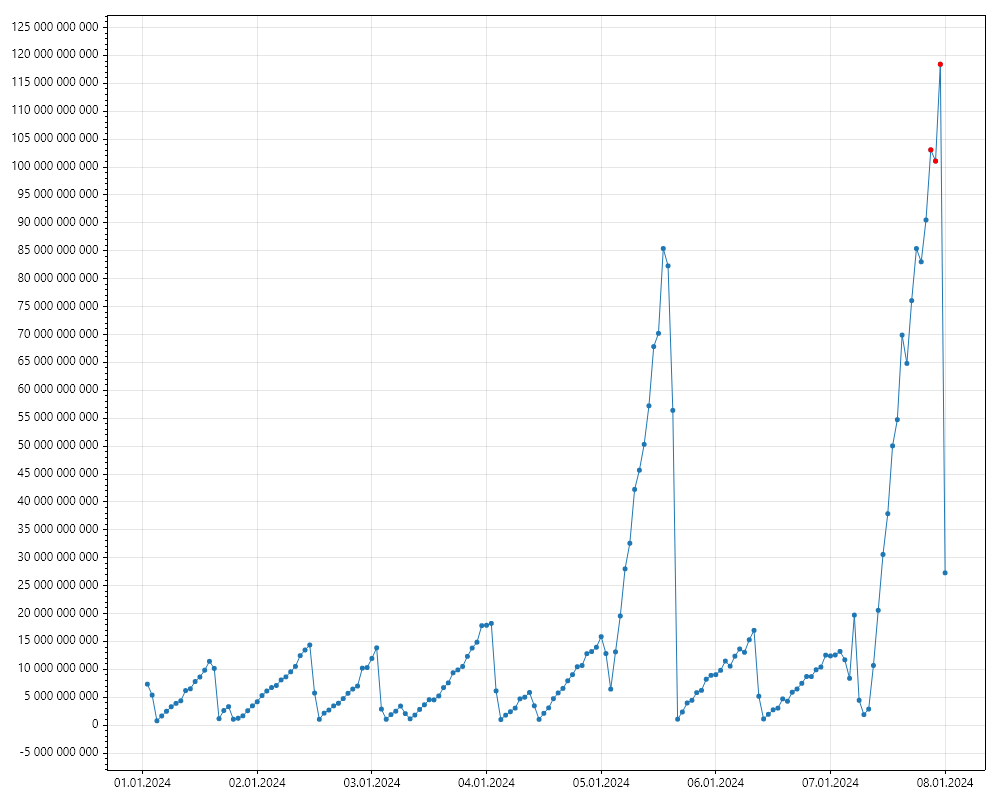


Рисунок 2 Общий объём полученного трафика для пользователя 5828

Аналогичные рисунку 2 - графики отправленного трафика и медианных значений – наблюдается скачок, который не соответствует нормальной активности пользователя, при этом число сессий и продолжительность не изменились.

Как мы видим по рисунку 1, продолжительность в норме, но объём входящего и исходящего трафика превышает норму, поэтому фиксируется аномалия.

Далее приведу график уровня доверия модели к действиям пользователя.

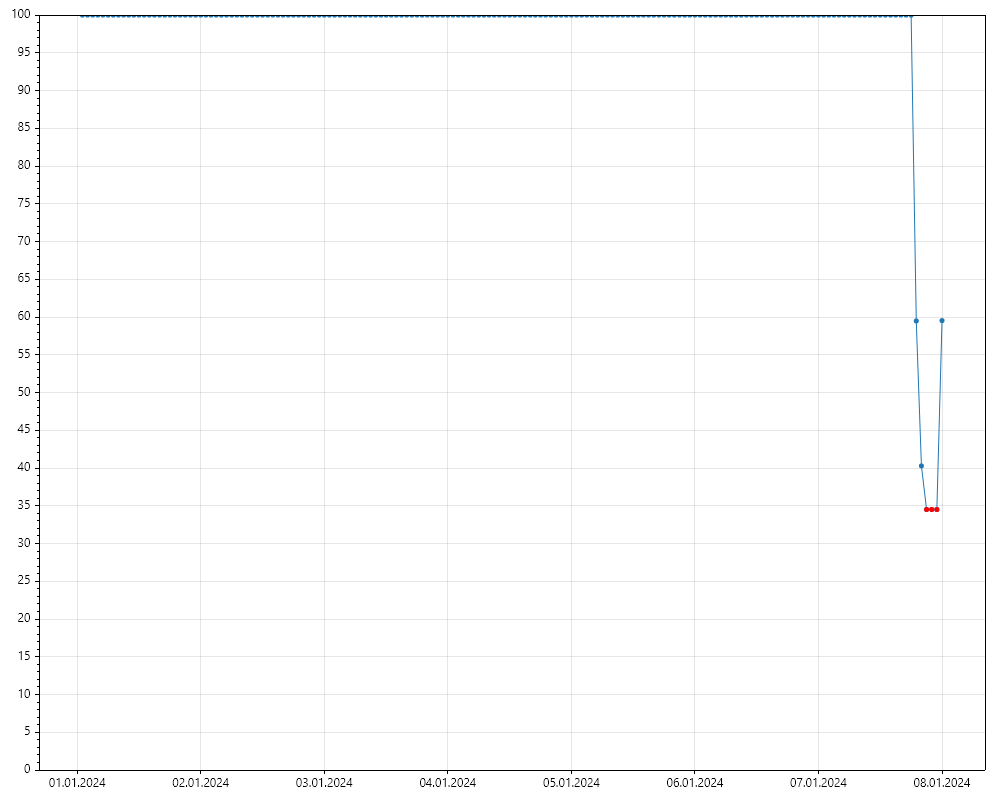


Рисунок 3 Изменение уровня доверия к действиям пользователя 5828

Вопрос: Почему более ранний аномальный пик, наблюдаемый на рисунке 2, не отражается на общем графике доверия как аномалия?

Ответ: При загрузке эталонных данных «странные» значения не отбрасывались, поэтому модель приняла их как фактические значения – нужно либо отбрасывать аномальные значения, но при этом сконфигурировать так, чтобы удалились явные отклонения, но остались пики, как на графике 1, либо исключать из общего датасета значения, где ранее были зафиксированы аномалии – всё это вопрос реализации политики в конкретной системе.

Анализ с точки зрения злоумышленника

В данном случае злоумышленник мог бы начать первую аномальную активность параллельно с какой-то аварией в системе - так, чтобы на его аномальное поведение внимание не обратили, а дальше держать уровень активность, аналогичной первой аномалии – это бы вписалось в портрет пользователя и его могли бы не заметить.

Альтернативный подход злоумышленника – медленными изменениями активности изменить портрет активности к такому, в котором он может реализовывать свои цели – общая уязвимость для аномальных алогоритмов.

Решать описанные проблемы нужно в конкретных случаях: задать модель изучаемых данных, которая коррелирует с анализируемыми аномалиями.

Применять индивидуальную политику в принятии решения об отбрасывании «странных» значений и уведомлении об аномалиях.

Итог по TelecomX

Полученная библиотека легко применима к изучаемому датасету TelecomX, но полученный алгоритм анализа можно обойти, уязвимости могут скрываться как в неадекватной политике, так и в неточных или недостаточных входных данных.

Итоги

Удалось построить библиотеку для анализа произвольного датасета с использованием базовых анализаторов, применить полученный инструмент на датасете TelecomX. Анализ результатов показал, что текущая реализации вполне подходит для быстрого и эффективно выявления аномалий в работе специализированных систем, изменения в которых почти не происходят – сервера. Для анализа пользователей применимо, но с допущениями.

Что дальше?

Применить полученный инструмент в анализе трафика конкретной машины, трафик перехватывать с помощью сниффера, например, WireShark, который уже умеет агрегировать трафик и собирать статистические параметры – их останется проанализировать. В результате анализа конкретной машины построить систему, которая выявляет нехарактерную сетевую активность.

Вероятно, потребуется принимать более продвинутые алгоритмы машинного обучения в построении базовых анализаторов, например, алгоритмы EllipticEnvelope или RandomForest.

В будущем нужно будет расширить подход в сторону анализа использования приложений, изменения файлов и журналов.

Но самое сложное - изучить, как особенности конкретного человека выражаются в активности его системы, как эти особенности выявлять и анализировать - это сложный, но перспективный вопрос для решения которого потребуется изучить психологическую, социальную и программную составляющие.

Возможно, этот проект станет шагом к лучшему пониманию человеческой природы, поможет найти ответ на вопрос как мы мыслим, и как принимаем решения – без этого не выйдет эффективно решить поставленные выше задачи.

Но продвинувшись в их решении, можно будет сделать работу всех пользователей более безопасной – повышая уровень «гигиены» отдельной системы, мы повышаем уровень общей безопасности, как сетевой, так и социальной – контроль за девиантным поведением отдельных людей через анализ аномалий в работе их систем.

Литература

1. Секреты и ложь. Безопасность данных в цифровом мире для блока литература, [Шнайер Брюс](https://readli.net/avtor/SHnayer-Bryus/), 2003
2. <https://selectel.ru/blog/ips-and-ids/>
3. <https://www.securityvision.ru/blog/obnaruzhenie-i-predotvrashchenie-vtorzheniy/>
4. <https://cisoclub.ru/metody-vyjavlenija-i-reagirovanija-na-anomalii-v-kompjuternyh-setjah/>
5. «Временные ряды в прогнозировании спроса, нагрузки на КЦ, товарных рекомендациях и поиске аномалий», [kirillkosolapov](https://habr.com/ru/users/kirillkosolapov/), https://habr.com/ru/articles/477206/
6. «9 подходов для выявления аномалий», [kirillkosolapov](https://habr.com/ru/users/kirillkosolapov/) <https://habr.com/ru/articles/477450/>
7. «Рекомендательные системы: проблемы и методы решения. Часть 1», [Jaylla](https://habr.com/ru/users/Jaylla/) <https://habr.com/ru/companies/prequel/articles/567648/>
8. «Открытый датасет TelecomX»

https://habr.com/ru/companies/arenadata/articles/856366/

1. «Образцы баз данных AdventureWorks» <https://learn.microsoft.com/ru-ru/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms>

https://data.mendeley.com/datasets/3t6rbtcms8/1

1. <https://learn.microsoft.com/ru-ru/dotnet/machine-learning/how-to-guides/getting-started-dataframe>
2. <https://habr.com/ru/articles/714988/>
3. https://scottplot.net/

На будущее, сравнить с известными системами

SNORT

Перехват и обработка сетевого трафика.

Типы сетевых атак. Подходы к их выявлению.

Построение алгоритма выявление небезопасной сетевой активности на основе анализа статистических данных по многим факторам.

Реализация гибкой настройки с добавлением своих данных и выбором параметров для возможности использования в узкоспециализированных системах.

https://selectel.ru/blog/ips-and-ids/

<https://www.icmm.ru/uchebnaya-deyatelnost/lektsii/514-ids>

<https://cyberleninka.ru/article/n/obzor-otkrytyh-naborov-dannyh-dlya-vyyavleniya-atak-na-veb-prilozheniya/viewer>

О том, где искать дата сеты

https://habr.com/ru/companies/vasexperts/articles/790192/

Идеи для реализации

1.Обязательно учесть систему доверенного третьего лица, которое будет реализовывать соответствующий интерфейс.

2.Файловая система – это модуль системы, которых может быть много, это обеспечить высокую пропускную способность, увеличивая нагрузку на сеть

Например: в Mongo DB существуют коллекции, было бы круто равномерно распределить все файлы на несколько «узлов файловых систем», каждый из которых отвечает за свою коллекцию, чтобы можно было параллельно обрабатывать большие потоки данных.

3.Существуют различные системы для хранения файлов на серверах – рассмотреть возможность создания прослойки для работы с такими системами как с будто бы это файловая система нашей машины – это позволит использовать уже существующие решения для создания высокопроизводительной, адаптивной под задачи, системы.

4.Определиться, как рассматривать ОС – пока я представляю ОС как множество сетевых узлов, на которых может быть любая Физическая ОС, подобранная для конкретной функции – т.е. рассматривать ОС как сборку уже готовых решения, на данный момент трудно будет заставить всех перейти на новую структуру

5.В последствии рассмотреть вариант создания своего микроядра, совместимого со всеми аппаратными решениями – это очень сложно(

6.Что делать с графикой? Видеокарта – тоже модуль системы. Полагаю, нужно посылать вычислительные задачи на разные узлы, чтобы успевать обрабатывать запросы графики.

7.Пользовательский компьютер – терминал или очень слабое, в сравнении с системой, устройство для примитивных задач.

8.В данной топологии ОС, которую хочу строить будет мало уязвимостей современных ПК, но не забывать про сетевые атаки – всё взаимодействие идёт по сети, не факт, что локальной.

9.Обязательно систему хранения логов, например, ElasticSearch и системы выявления аномалий(Алгоритмическую и AI)

10.В системе скорее всего будут важные распределительные пункты, они должны быть просты, их цель распределить задачи по распределённым узлам.

11.Поговорить о безопасности(Литература 4 – интересная статья)

12.Сервер реинкарнации – от него порождаются все процессы, он отлавливает подвисшие-зомби процессы, удаляет их, перезапускает и сообщает об этом другим серверам.

13.Возможность склонировать и запустить параллельно уже работающий модуль системы для того, чтобы распараллелить нагрузку или чтобы удобно протестировать без вреда для всей системы.